SINTONIA ADAPTATIVA DE CONTROLADORES PID UTILIZANDO OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS/MAPAS CAÓTICOS

Larissa A. Baratieri* Fábio R. Durand* Cristiano M. Agulhari* Vinícius Suterio*
Bruno L.G. Costa* Bruno A. Angélico†

*Avenida Alberto Carazzai, 1640
Laboratório de Simulação e Otimização Computacional
Universidade Tecnológica Federal do Paraná (UTFPR)
Cornélio Procópio, Paraná, Brasil

†Av. Prof. Luciano Gualberto - Travessa 3, nº 158
Laboratório de Automação e Controle
Escola Politécnica da Universidade de São Paulo (POLI-USP)
São Paulo, São Paulo, Brasil

Emails: lbaratieri@alunos.utfpr.edu.br,{fabiodurand,agulhari}@utfpr.edu.br, vinicius.suterio@yahoo.com.br, brunolgcosta@gmail.com, angelico@lac.usp.br

Abstract— This paper shows the comparison of two methods used to solve the tuning of a PID controller. The tuning of this controller is not always so trivial and easy to find, with many tabulated methods being used and requiring a longer time to perform the calculations. To optimize this problem, computational methods are used to find an optimized tuning response on a considerable computational time scale. The objective of this work is the comparison of two computational metaheuristic tools: the Particle Swarm Optmization (PSO) and the Chaotic Particle Swarm Optimization (CPSO). The comparison of these two will show the efficiency of the CPSO in relation to the PSO, showing that it is able to correct some defects of the PSO in relation to being stuck in the minimum local of the function and thus managing to find the response of the function in relation to all its domain (global minimum). In addition, this work makes use of two cost functions, one usually used and another one proposed by this paper, and defining which one will have a better answer regarding the stability of controller gains and the response of the plant.

Keywords— Particle Swarm Optmization, Chaotic Particle Swarm Optimization, tuning, PID, control action.

Resumo— Este artigo mostra a comparação de dois métodos utilizados para resolver a sintonia de um controlador PID. A sintonia de tal controlador nem sempre é tão trivial e fácil de se achar, sendo que muitos métodos tabelados são utilizados e requerem um tempo maior para a realização dos cálculos. Para otimizar esse problema, são utilizados métodos computacionais para encontrar uma resposta otimizada de sintonia em uma escala de tempo computacional considerável. O objetivo deste trabalho é a comparação de duas ferramentas computacionais metaheurísticas: o *Otimização por Exame de Partículas* (PSO) e o *Otimização por Exame de Partículas Caótico* (CPSO). A comparação desses dois mostrará a eficiência do CPSO em relação ao PSO, mostrando que aquele é capaz corrigir alguns defeitos do PSO em relação a ficar limitado aos mínimos locais da função, e assim conseguindo achar a resposta da função em relação a todo seu domínio (mínimo global). Além disso, o trabalho faz a utilização de duas funções custo, uma usualmente usada e outra proposta por esse artigo, e definindo qual terá uma melhor resposta em relação as estabilidades dos ganhos do controlador e a resposta da planta.

Palavras-chave— Otimização por enxame de partículas, Otimização por enxame de partículas caótico, sintonia, PID, ação de controle.

1 Introdução

O controlador Proporcional Integrativo Derivativo, PID, é uma técnica de processo que tem sido utilizada em indústrias de sistemas de controle devido ao seu desempenho robusto e praticidade no manuseio. Entretanto, a sintonia desse controlador pode ainda ser considerada um desafio, alguns o ajustam pelo método de tentativa e erro e outros utilizam os métodos convencionais, tais como: método de Ziegler-Nichols, método de Cohen-Coon, método de Chien-Hrones-Reswick (CHR), entre outros (Astrom and Hägglund, 1995). Esses métodos são calculados a partir de tabelas e nem sempre garantem uma sintonia desse controlador.

Em muitas situações, os controladores PID são substituídos por controladores Proporcional-Integral, PI, em que a parte de ação derivativa é retirada, sacrificando o desempenho e a eficiência da operação, por um processo rápido e fácil de sintonizar (Torres et al., 2017).

Diante desse contexto, devem-se utilizar métodos mais eficazes que possam garantir uma sintonia satisfatória. Os métodos de otimização podem ser a resposta para tal problemática, uma vez que esses algoritmos são eficazes para problemas computacionais. Esses métodos se adaptam ao problema e por conseguinte buscam e convergem para soluções consideradas como ótimas durante um tempo computacional aceitável.

Diferentes métodos de otimização são utilizados para a sintonia desse controlador. Pode-se

citar como exemplo a Otimização por Enxame de Partículas, PSO (Particle Swarm Optimization), Evolução Diferencial, DE (Differencial Evolution), entre outros.

Em Yanzi Miao et al. (2015) é utilizado o PSO para a sintonização de um controlador PID. Nesse trabalho há o melhoramento do algoritmo para encontrar os ganhos do controlador.

Já em de Andrade et al. (2013) há a comparação entre os métodos tradicionais por tabelamento e o PSO, provando que o método computacional, ou seja o PSO, é melhor para a sintonização.

Em Senkerik et al. (2014) há uma comparação do algoritmo DE em relação ao PSO para a sintonia de um controlador PID, sendo adicionado, em cada algoritmo, o Gerador de Número Pseudoaleatório Criptograficamente Seguro (CPRNG) gerando assim um aspecto caótico e, consequentemente, são ampliadas as buscas do domínio dos algoritmos . Em Nie et al. (2016) utiliza uma Otimização Adaptativa de Enxame de Partículas do Caos (ACPSO) para a sintonização de um controlador PID, mostrando que essa melhoria sobre o PSO é capaz de resultados robustos e eficientes.

Com o intuito de facilitar o processo de trabalho das indústrias em sintonizar o controlador, até mesmo poupando tempo de trabalho calculando os ganhos do controlador, esse estudo apresenta uma aplicação do algoritmo CPSO, na sintonia de um controlador PID da malha de nível de uma planta didática, sendo que esse método será comparado ao algoritmo PSO para verificar a eficácia de um em relação ao outro ao determinar os parâmetros do controlador.

Além disso, esse estudo aplica uma função custo diferente dos trabalhos citados anteriormente, os quais utilizam apenas os índices de desemprenho do controlador (Nie et al., 2016). Essa função custo tem por objetivo corrigir o erro sobre a planta de uma forma a garantir os melhores valores possíveis para os ganhos garantindo a estabilidade do sistema.

Os dados utilizados nesse projeto foram obtidos de uma planta didática que foi alvo de estudo por (de Andrade et al., 2013). Eles serviram de base para comparação entre os algoritmos, mostrando a capacidade do CPSO convergir de uma forma mais eficaz para a sintonia do controlador na planta.

A organização do restante do artigo está como se segue. A Seção 2 abordará alguns conceitos da fundamentação teórica, a Seção 3 abordará a metodologia e efetuará uma análise do desempenho, a Seção 4 apresenta os resultados obtidos e a Seção 5 conclui o estudo.

2 Fundamentação Teórica

2.1 Planta

A planta de controle e os seus respectivos dados foram obtidos de de Andrade et al. (2013). Trata-se de uma planta didática industrial modificada da fabricante FESTO, a qual possui quatro malhas de controle: temperatura, pressão, nível e vazão (Almeida, 2012).

A planta pode ser vista na Figura 1, em que o *CompactRIO* é o controlador da planta, a bomba centrífuga é o atuador do sistema de controle e a válvula de distúrbio é responsável por inserir distúrbios no sistema.

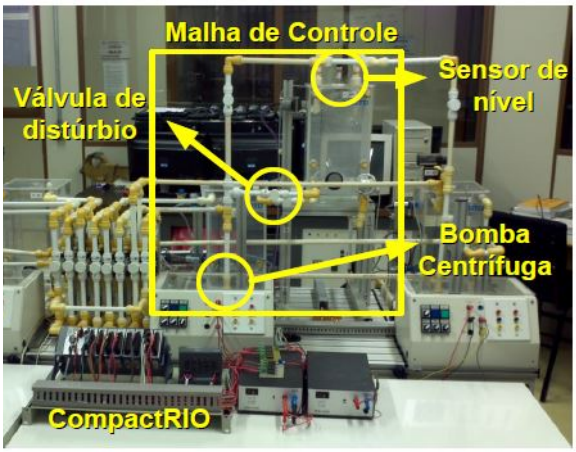

Figura 1: Planta da malha de controle.

A função de transferência dessa malha em sintonia com o controlador PID é dada por:

$$G(s) = \frac{2.677e^{-9.41s}}{75.29s + 1} \tag{1}$$

2.2 Particle Swarm Optimization PSO

O método PSO trata de uma metaheurística baseada nos padrões de comportamento da natureza. As partículas consideradas pelo algoritmo se comportam como pássaros à procura de alimento ou do local de seus ninhos, utilizando o aprendizado próprio (partículas) e o aprendizado do bando (enxame) trocando informação entre eles. O algoritmo de PSO engloba conceitos simples requerendo apenas operadores matemáticos simples. Apesar de classificado como evolucionário, não apresenta a característica de sobrevivência do mais apto ou a utilização de operadores genéticos como o cruzamento e a mutação.

No PSO tem-se um espaço delimitado de busca. Neste espaço busca-se uma solução candidata a partir de uma população de soluções candidatas, denominada partículas, fazendo-se uso de parâmetros matemáticos que determinam a posição e a velocidade de tais partículas.

O movimento de cada partícula depende da sua melhor posição encontrada e a da posição global do restante das partículas. O movimento das partículas é distribuído aleatoriamente no espaço, cada uma possuindo uma posição e velocidade individuais. A velocidade proporciona o deslocamento da partícula em busca de um melhor desempenho (Rao, 2009).

$$p_{id}[t] = p_{id}[t-1] + v_{id}[t-1] \tag{2}$$

Sendo $p_{id}[t]$ a posição no instante t e $v_{id}[t-1]$ a velocidade da partícula no instante $t-1$.

A expressão do PSO é dada por:

$$v_{id}[t+1] = w \cdot v_{id}[t] + \Phi_1 \cdot rand_{id}(p_{id}^{best}[t] - p_{id}[t])$$

$$+\Phi_2 \cdot rand_{id2}(p_{gd}^{best}[t] - p_{id}[t]) \tag{3}$$

Em que w é o coeficiente de inércia, Φ_1 é o quanto a partícula confia em sua própria experiência e Φ_2 é o quanto a partícula confia em sua vizinhança. Esses coeficientes podem ser obtidos por intermédio de conhecimento empírico ou pelo conhecimento de detalhes do problema em que se está trabalhando.

O conhecimento da melhor posição encontrada por qualquer partícula no conjunto é adicionada à velocidade de todas as partículas, criando um componente que irá na direção desta partícula.

Se $p_{id} > p_{gd}^{best}[t]$ então $v_{id}[t] = v_{id}[t-1] - rand \cdot \Phi_2$

Se $p_{id} < p_{gd}^{best}[t]$ então $v_{id}[t] = v_{id}[t-1] + rand \cdot \Phi_2$

A função *rand* gera os valores pseudo-aleatórios normalizados entre [0,1]. E Φ_2 é a constante de intensidade de deslocamento (melhor posição global).

O cálculo da velocidade da melhor posição encontrada pela partícula individual é dada por:

Se $p_{id} > p_{gd}^{best}[t]$ então $v_{id}[t] = v_{id}[t-1] - rand.\Phi_1$

Se $p_{id} < p_{gd}^{best}[t]$ então $v_{id}[t] = v_{id}[t-1] + rand.\Phi_1$

Em que Φ_1 é a constante de intensidade de deslocamento (melhor posição da partícula).

A Figura 2 apresenta o fluxograma de implementação do algoritmo (Eberhart et al., 1996). Cada um dos passos serão explanados a seguir.

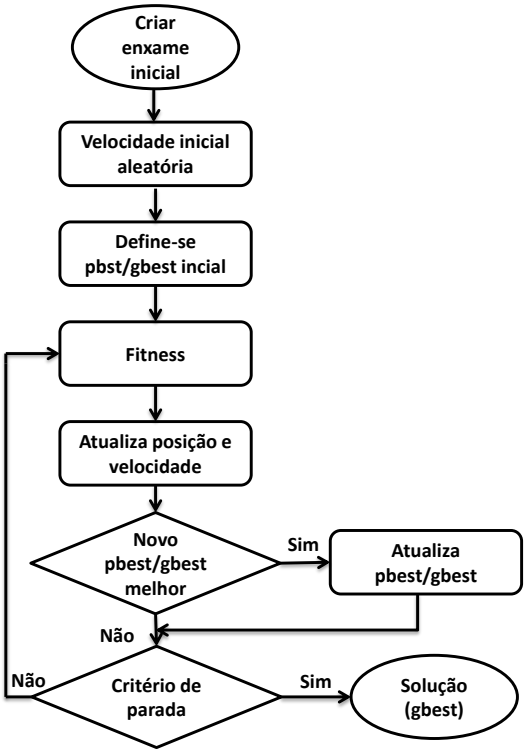

Figura 2: Algoritmo do PSO.

1. Inicializar a população de partículas com posições e velocidades aleatórias no espaço D dimensional;
2. Avaliar a aptidão de cada uma das partículas;
3. Comparar o valor obtido da partícula i com *pbest*. Se o valor for melhor, atualizar *pbest* com o novo valor;
4. Comparar o valor obtido com o melhor valor global *gbest*. Se for melhor, atualizar *gbest* com o novo valor;
5. Atualizar a velocidade da partícula;
6. Atualizar a posição da partícula;
7. Repetir os passos 2-6 até que algum critério de parado seja alcançado. Neste caso o critério de para é o número de épocas do algoritmo, ou seja, o número de vezes que o algoritmo deverá ser compilado.

2.3 Chaotic Paticle Swarm Optimization - CPSO

O CPSO é um método de otimização que consiste em uma melhoria do algoritmo PSO. Tal método

utiliza agentes caóticos para ampliar as áreas de busca desse algoritmo.

Utilizando como base a Teoria do Caos (Lorenz, 1993), o CPSO é um método que tem como base os mapas caóticos os quais são feitos a partir de equações matemáticas de sistemas dinâmicos que são altamente sensíveis as condições iniciais. Pode-se citar como exemplo o mapa caótico logístico (May, 1976), considerado um dos mapas mais simples em relação aos demais existentes. O fundamento desse mapa é uma Equação 4, que descreve o crescimento populacional

$$x_{n+1} = r x_n (1 - x_n), \tag{4}$$

em que x_n à população de iteração n e r é o potencial biótico. Dependendo do valor de r a população pode atingir diferentes estágios: estável, extinta ou caótico.

Para esse trabalho, é necessário que a população descrita atinja o estágio caótico, portanto foi escolhido uma valor para r igual 4.

A Figura 3 representa uma parte do mapa caótico logístico, em que os valores dessa função foram normalizados entre 0 e 1.

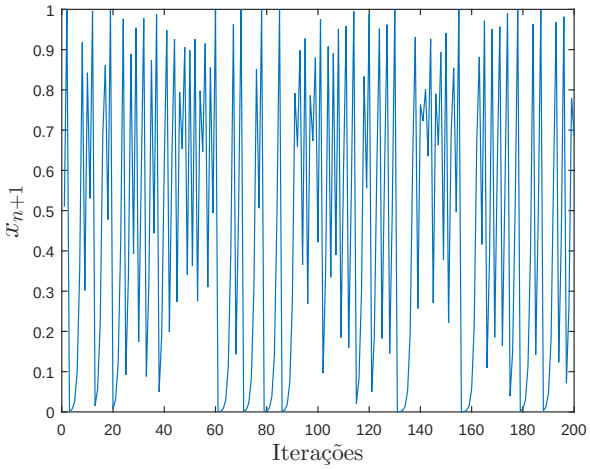

Figura 3: Mapa caótico logístico.

A utilização da dinâmica caótica no PSO surgiu para corrigir a área de busca desse algoritmo. Ampliando essa área, o algoritmo não fica preso nos mínimos locais, ele possui uma ampla região de busca e assim podendo achar a resposta global em uma quantidade menor de iterações.

3 Análise de Desempenho

A dificuldade de sintonizar um controlador está na obtenção dos ganhos. Portanto os algoritmos foram utilizados nessa parte do estudo.

A Figura 4 representa o digrama de blocos da malha de controle, em que $G_c(s)$ é o controlador PID e $G_P(s)$ é o objeto controlado. O bloco PSO/CPSO é o algoritmo que será trabalhado sobre os parâmetros de ganho do controlador.

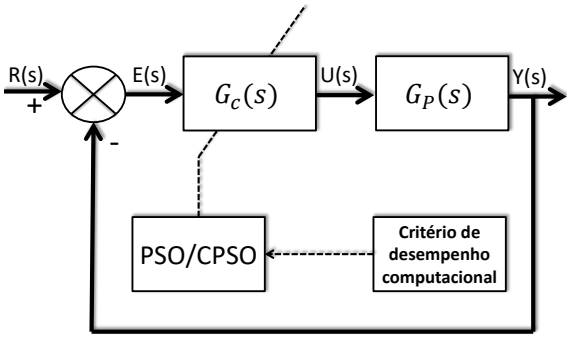

Figura 4: Diagrama de blocos da malha de controle.

Os dois algoritmos foram construídos utilizando o software MATLAB®. Para isso deve-se definir uma função custo para a utilização dessas metaheurísticas. Tal função, representado pela Função 5, depende de duas parcelas de minimização: uma baseada no índice desempenho ITAE (Integral of Time multiplied by Absolute of the Error) que é uma medida quantitativa do desempenho de um sistema indicado para reduzir a contribuição de grandes erros inicias no valor da integral de desempenho. E a outra parcela dedicada a minimizar a ação de controle ($u_c(t)$) do sistema, por meio do cálculo da variância do vetor da ação de controle:

$$F_{custo} = W_1 \left(\int_0^T t|e(t)|dt \right) + W_2 var(u_c(t)). \tag{5}$$

As ponderações, que são representadas por W_1 e W_2, são atribuídas a cada parcela da equação.

A Tabela 1 contém os parâmetros usados na implementação dos algoritmos.

Tabela 1: Parâmetros CPSO.

Parâmetros	
Iterações	30
População	20
Coeficiente de Inércia	0.75
Φ_1 e Φ_2	2
Velocidade máxima	5
Velocidade mínima	-5
W_1	1
W_2	2

A implementação do algoritmo CPSO foi realizada da mesma forma que a implementação do algoritmo PSO, porém ao invés de usar a função aleatória *rand* do MATLAB®, substituiu-a por linhas de código utilizando o mapa caótico logístico.

4 Resultados

Para garantir uma melhor comparação entre os procedimentos utilizados, foram realizadas cinco repetições com cada algoritmo, sendo que a melhor reposta obtida será apresentada com os gráficos: convergência dos ganhos, resposta de degrau e ação de controle. As demais iterações serão analisadas somente com o gráfico da convergência da função custo.

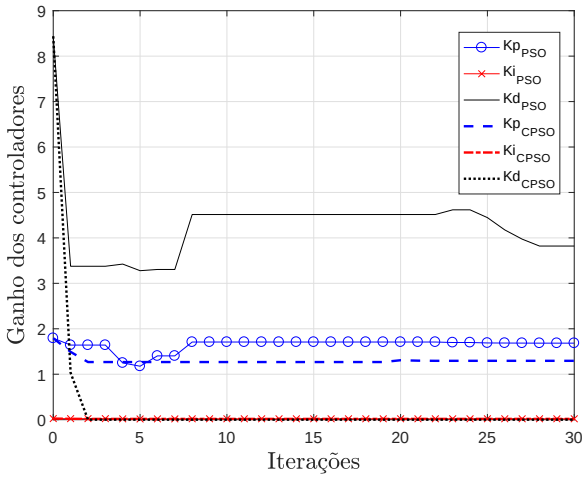

Figura 5: Convergência dos Ganhos.

O gráfico da Figura 5 mostra os ganhos proporcional (Kp), derivativo (Kd) e integrativo (Ki) de cada algoritmo simulado (PSO e CPSO), nota-se que convergência do ganho derivativo do CPSO foi mais rápida que do PSO.

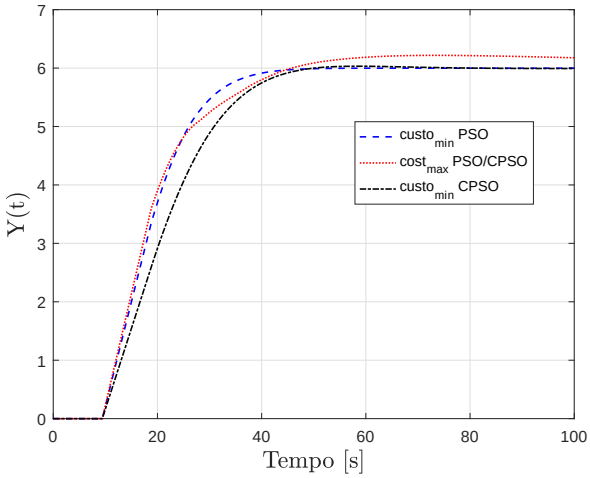

Figura 6: Resposta Degrau.

O gráfico da Figura 6 correspondente à convergência do degrau unitário dos dois algoritmos utilizados, que possuem uma diferença mínima em relação ao custo mínimo ($custo_{min}$) da reposta degrau. Já em relação ao custo máximo ($custo_{max}$), os dois possuem a mesma curva.

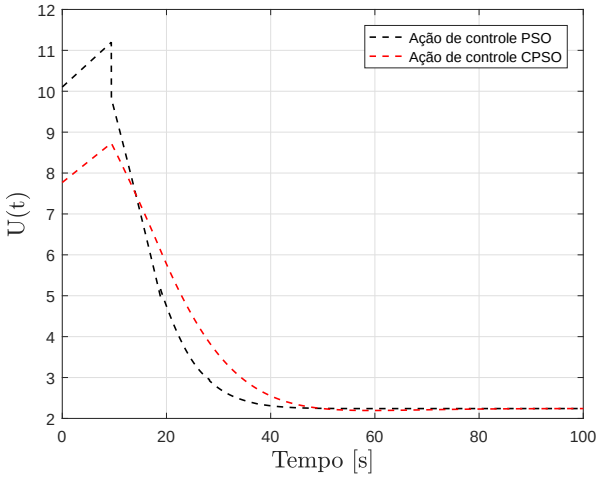

Figura 7: Ação de Controle.

Na Figura 7 pode-se averiguar que a ação de controle do CPSO, representado pela linha vermelha, é menor que o PSO, representado pela linha preta.

O gráfico representado na Figura 8 mostra à convergência do resultado da função custo, em que é visível que a resposta dada pelo CPSO é melhor do que o PSO, pois o CPSO não ficou preso em mínimos locais, portanto ele conseguiu achar o melhor global em um número menor de iterações. Já o PSO ficou preso diante de vários mínimos locais até achar o mínimo global, porém isso demorou mais iterações.

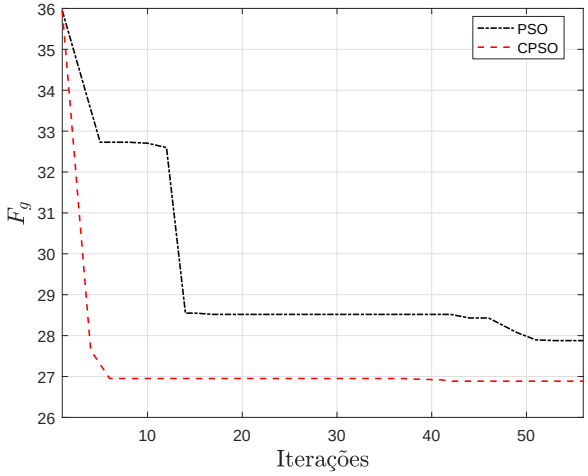

Figura 8: Convergência da função custo.

As Tabelas 2 e 3 mostram as demais simulações feitas com os algoritmos PSO e CPSO, respectivamente. Elas indicam em que momento os

algoritmos atingiram um valor próximo do esperado, e em qual iteração esse valor foi atingido.

Tabela 2: Simulações PSO.

PSO		
Simulação	**Função Custo**	**Iteração**
1	26.9864	25
2	26.9199	4
3	27.8780	30
4	26.7562	17
5	26.7562	17

Tabela 3: Simulações CPSO.

CPSO		
Simulação	**Função Custo**	**Iteração**
1	26.8238	16
2	26.9047	3
3	26.9505	2
4	26.8983	3
5	26.8933	3

O número máximo de iterações realizadas pelos algoritmos foi 30, tanto para o PSO como para o CPSO. Pode-se observar que o CPSO convergiu com menos iterações.

Além disso, deve-se levar em consideração que função custo proposta pelo trabalho gerou os gráficos representados pelas Figuras 5, 6, 7 e 8. Já os gráficos representados pelas Figuras 9, 10 e 11 correspondem a função custo citado em (Nie et al., 2016), função a qual possui somente a parcela ITAE em seu cálculo.

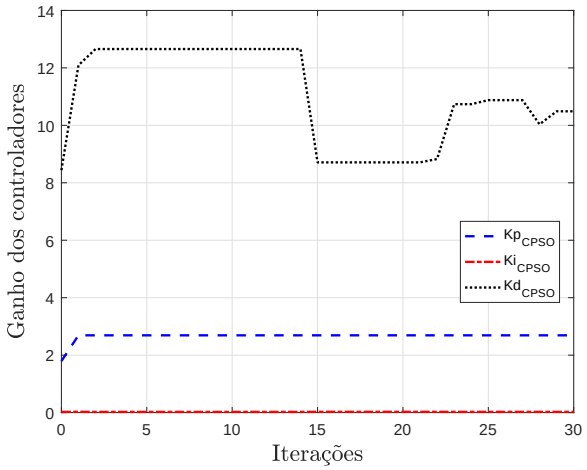

Figura 9: Convergência dos Ganhos.

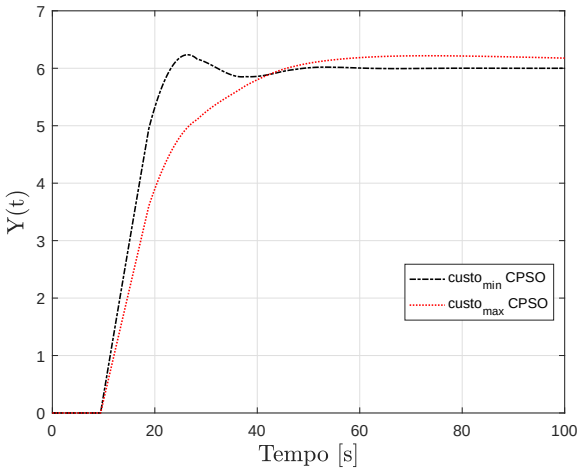

Figura 10: Resposta Degrau.

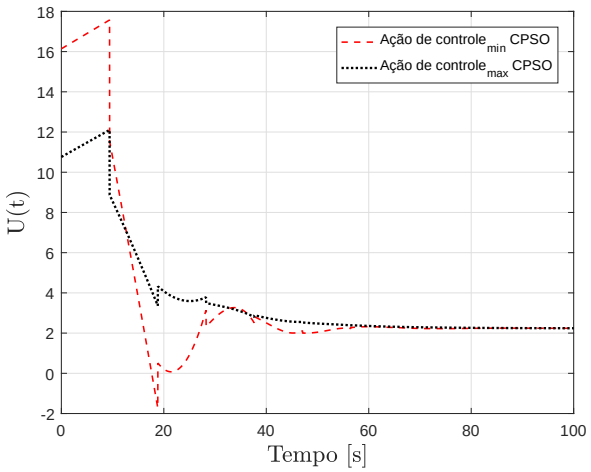

Figura 11: Função custo: ação de controle.

Verifica-se que os ganhos não convergiram de uma forma que garantisse uma estabilidade direta no controlador, isso é refletido no gráfico que mostra a curva da resposta degrau, que antes de convergir possui um *overshoot* indicando que a planta sofreu oscilações durante o período de transitório, atingindo a estabilidade no regime permanente.

Já a função custo proposta pelo trabalho, e apresentada na Figura 6, o qual utiliza a variância da ação de controle mais o ITAE, mostra a convergência dos ganhos garantindo que a resposta degrau atinge a estabilidade de uma forma mais suave, sem *overshoot*, não havendo indícios de oscilações até atingir o regime permanente.

5 Conclusões

A proposta desse trabalho foi mostrar a eficiência do CPSO sobre o PSO para sintonizar os parâmetros de um controlador PID e apresentar uma função custo que estabilize o controlador de uma forma mais rápida e melhorada. Sendo assim, o CPSO mostra-se mais eficiente para a sintonização do controlador PID, apresentando maior pre-

cisão e eficácia devido ao seu tempo computacional. Além disso sua eficiência também se dá pelo fato de possuir uma convergência para a solução global em uma quantidade menor de iterações, não ficando muito preso em soluções locais, assim não possui uma convergência prematura tornando-se mais robusto.

A função custo proposta por este estudo converge os ganhos do controlador de uma forma melhor, com isso o sistema atinge a estabilidade sem *overshoot*.

Referências

Almeida, J. P. L. S. d. (2012). *Automação de uma Planta Didática de Sistemas de Controle, 122f.*, Trabalho de Conclusão de Curso (Graduação) | Curso Superior de Tecnologia em Automação Industrial. Universidade Tecnológica Federal do Paraná: Cornélio Procópio.

Astrom, K. J. and Hägglund, T. (1995). *PID Controllers: Theory, Design, and Tuning*, 2 edn, Instrument Society of America, Research Triangle Park.

de Andrade, L. H. S., Costa, B. L. G. and Angélico, B. A. (2013). *PSO Aplicado À Sintonia do Controlador PI/PID da Malha de Nível de uma Planta /didática Industrial*, Sbai - Simpósio Brasileiro de Automação Inteligente.

Eberhart, R., Simpson, P. and Dobbins, R. (1996). *Computational Intelligence PC Tools*, MA Academic Press Professional, Boston.

Lorenz, E. N. (1993). *The Essence of Chaos*, UCL Press. 227 pp.

May, R. M. (1976). Simple mathematical models with very complicated dynamics, *Nature* **261**(5560): 459.

Nie, S.-K., Wang, Y.-J., Xiao, S. and Liu, Z. (2016). *An adaptive chaos particle swarm optimization for tuning parameters of PID controller*, Reading, MA.

Rao, S. (2009). *Engineering Optimization: Theory and Practice*, Fourth Edition. New Jersey.

Senkerik, R., Pluhacek, M., Zelinka, I., Davendra, D. and Oplatkova, Z. K. (2014). *Comparison of Chaos Driven PSO and Differential Evolution on the Selected PID Tuning Problem*, HAL.

Torres, W. L., Araujo, I. B. Q., Filho, J. B. M. and Junior, A. G. C. (2017). *Mathematical Modeling and PID Controller Parameter Tuning in a Didactic Thermal Plant*, IEEE Latin America Transactions.

Yanzi Miao, Y., Liu, Y., Chen and Jin, H. (2015). *PID Controller Parameter Tuning Based on Improved Particle Swarm Optimization Algorithm*, International Conference on Electrical, Electronics and Mechatronics.